# SAR Image Processing with Python

This repository contains a Jupyter Notebook that demonstrates Synthetic Aperture Radar (SAR) image preprocessing, filtering, and visualization using Python.  
It is designed as a simple, reproducible workflow for reading SLC (Single Look Complex) data, applying range-domain windowing, and comparing original vs. filtered SAR images.

## 📂 Contents

- sar\_processing.ipynb – main notebook with step-by-step processing

- requirements.txt – Python dependencies

- README.md – documentation (this file)

## ⚙️ Setup

1. Clone the repository:

git clone https://github.com/your-username/sar-processing.git  
cd sar-processing

2. Install dependencies with pip:

pip install -r requirements.txt

Or using Conda:

conda create -n sar-env python=3.11  
conda activate sar-env  
pip install -r requirements.txt

## ▶️ Usage

1. Launch Jupyter:  
 jupyter notebook

2. Open sar\_processing.ipynb in your browser.

3. Run the cells sequentially:  
 - Preprocessing  
 - Windowing & FFT filtering  
 - Visualization

## 📊 Features

* Load complex SAR data (.nitf) with SARPy
* Apply Hanning window in range domain
* Perform FFT-based filtering
* Display original vs. filtered SAR images in linear or dB scale
* Compare full image and zoomed-in regions
* Support for adaptive contrast scaling (mean ± std)

## 📁 Data

This repository does not include SAR data files due to size restrictions.  
Expected input format: .nitf (NGA SICD-compliant)  
Example filename: IMG-VV-STRIXB-20220811T004713Z-SMSLC-SICD.nitf  
Place your SAR files in the same directory as notebook project.

## 📝 Notes

Tested with Python 3.11, numpy, matplotlib, scipy, sarpy  
If using Conda, you may also export an environment.yml for reproducibility  
Make sure your system has enough memory (SAR .nitf files can be very large)

## 📜 License

This repository is for educational and demonstration purposes.  
Data and code should not be used for operational or restricted applications without permission.